Approssimazione di un equazione ai valori al contorno

col metodo upwind

Enrico Bertolazzi

Carica le librerie

```
> restart ;
  with(LinearAlgebra) :
  with(plots):
Warning, the name changecoords has been redefined
```

Definisce la procedura center

```
> center := proc(p, q, r, L::Vector, R::Vector, n::integer)
    local i::integer,
          j::integer,
          h::float,
          xi::float,
          alpha::float,
          beta::float,
          gamma::float,
          omega::float,
          A::Matrix,
          b::Vector,
          y::Vector,
          res:Vector;

    # alloca la matrice e i vettori
    A := Matrix(n-1,n-1,shape=band[1,1]) ;
    b := Vector(n-1) ;
    y := Vector(n-1) ;

    h := (R[1]-L[1])/n ;

    for i from 1 to n-1 do
      xi := L[1] + i*h ;
      alpha := 1-h*min(0,p(xi)) ;
      beta  := 1+h*max(0,p(xi)) ;
      gamma := -2+(h^2)*q(xi)-h*abs(p(xi)) ;
      omega := (h^2)*r(xi) ;
      # costruisce termine noto e sistema lineare
```

```
      b[i]   := omega ;
      A[i,i] := gamma ;
      if i > 1 then
        A[i,i-1] := alpha ;
      else
        b[i] := evalf(b[i] - alpha * L[2]) ;
      end if ;
      if i < n-1 then
        A[i,i+1] := beta ;
      else
        b[i] := evalf(b[i] - beta * R[2]) ;
      end if ;
    end do;

    print(A,b) ;

    # risolve il sistema lineare
    y := evalf(LinearSolve(A,b)) ;
    print(y) ;

    # costruisce la lista per la visualizzazione
    res := [] ;
    for i from 1 to n-1 do
      xi  := L[1] + i*h ;
      res := [ op(res), [xi, y[i]] ] ;
    end do;
    res := [ convert(L,list), op(res), convert(R,list) ] ;

    return res ;
  end proc :
```

Esempio d'uso

```
> # risolve il problema
  a := -100 :
  p := x -> a :
  q := x -> 0 :
  r := x -> a :
  L := <0,0> :
  R := <1,0> :
```

```
> pt1 := center(p,q,r,L,R,10) :
  pt2 := center(p,q,r,L,R,25) :
  pt3 := center(p,q,r,L,R,50) :
```

$$\begin{bmatrix} -12 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 11 & -12 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 11 & -12 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 11 & -12 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 11 & -12 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 11 & -12 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 11 & -12 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 11 & -12 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11 & -12 \end{bmatrix}, \begin{bmatrix} -1. \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1. \end{bmatrix}$$

$$\begin{bmatrix} 0.099999999610000000 \\ 0.199999995399999990 \\ 0.299999948700000018 \\ 0.399999435600000008 \\ 0.499993790800000004 \\ 0.599931698699999960 \\ 0.699248685200000052 \\ 0.791735537200000006 \\ 0.809090909100000034 \end{bmatrix}$$

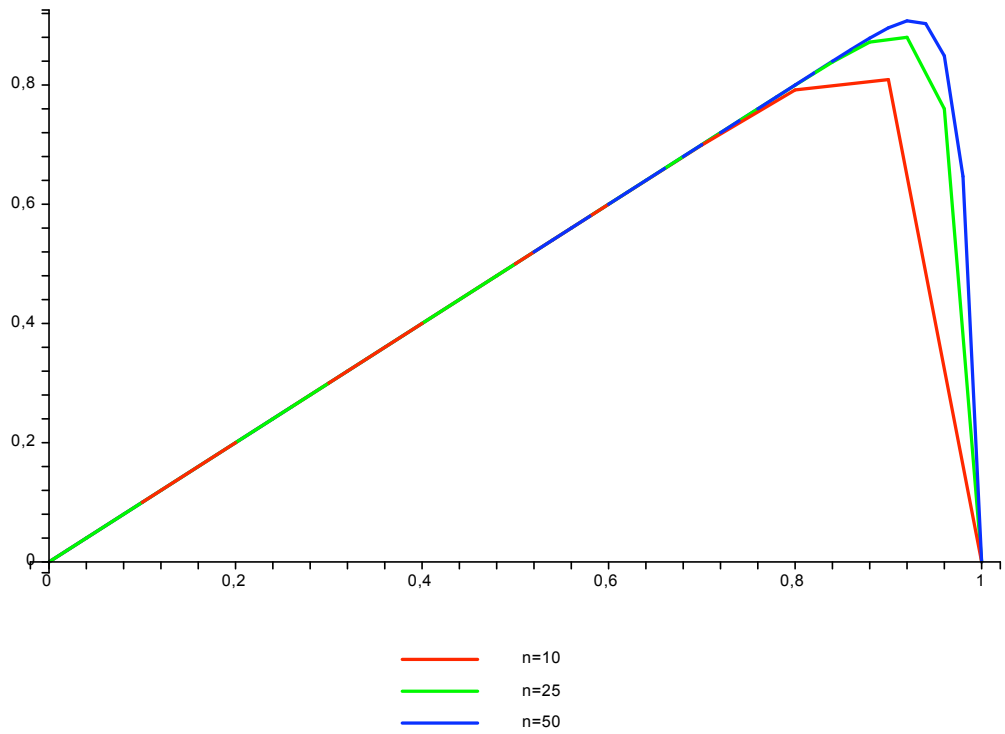
$$\begin{bmatrix} \text{24 x 24 Matrix} \\ \text{Data Type: anything} \\ \text{Storage: band[1, 1]} \\ \text{Order: Fortran\_order} \end{bmatrix}, \begin{bmatrix} \text{1 .. 24 Vector[column]} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

$$\begin{bmatrix} \text{1 .. 24 Vector[column]} \\ \text{Data Type: float[8]} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

$$\begin{bmatrix} \text{49 x 49 Matrix} \\ \text{Data Type: anything} \\ \text{Storage: band[1, 1]} \\ \text{Order: Fortran\_order} \end{bmatrix}, \begin{bmatrix} \text{1 .. 49 Vector[column]} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

$$\begin{bmatrix} \text{1 .. 49 Vector[column]} \\ \text{Data Type: float[8]} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

```
> # disegna le soluzioni
  plot([pt1,pt2,pt3],
       style=line,
       thickness=2,
       color=[red,green,blue],
       legend=["n=10","n=25","n=50"]);
```




```
>
```